Terimler

**Commit**, oyunlardaki checkpoint olarak düşünebilirsiniz.

**Branch**, ingilizcede “dal” anlamına gelir. Branch'ler oluşturularak bir projede birden fazla kişinin ekip halinde çalışması sağlanır.

**Gitignore**, git'in görmezden gelmesini istediğimiz klasörlerimiz veya dosyalarımız varsa kullanırız.

Ana branch'imiz **Master** veya **main** olarak isimlendirilir.

**HEAD**, bize git içerisinde bulunduğumuz konum,branch veya commit'i gösterir ama genelde son commit'imizi gösterir.

**Merge**, ingilizcede “birleştirmek” anlamına gelir. 2 branch'i birleştirmemize yarar.

**Private** dosyalarımızı,kodlarımızı vs. sadece biz ve bizim yetkilendirdiğimiz kişiler görebilir veya değişiklik yapabilir.

**Pull** proje ana dosyasındaki yaptığımız değişikliklerin bilgisayarımızdaki versiyonuna çekilmesini sağlar.

**Pull** **request** projen üzerinde değişiklik yaptım, sen de bu değişiklikleri onaylarak projene merge edebilirsin anlamına gelir.

**Fork** projeyi clone işlemindeki gibi kopyalarsın ama yollar repo'nun sahibi ile ayrılır yani push yaptığında sadece senin hesabındaki repo'da değişiklik olur.

KOMUTLAR

**pwd(print working directory)** --> içinde bulunduğun dosya yolunu gösterir.

**cd(change directory)** --> klasörler arasında gezinmemizi sağlar.

**cd ..** --> bir önceki klasöre geri gel demek.

**clear** --> terminali temizler

**mkdir(make directory)** --> bulunduğun dosya yolunda klasör oluşturur.

**touch dosya\_adi.uzanti** --> bulunduğun dosya yolunda dosya oluşturur.

**rm(remove) dosya\_adi.uzanti** --> dosya kaldırır. rm komutu sadece dosyaları siler.

**rm -rf klasor\_adi** --> klasör siler

**git config --global user.name "Kadir Akipek"** --> kullanıcı adımızı girmemizi sağlar, global'i siler ve adımızı yazmadan enter yaparsak sistemde bulunan bilgiyi gösterir.

**git config --global user.email akipekkadir23@gmail.com** --> gmailimizi girmemizi sağlar, global'i siler ve mailimizi yazmadan enter yaparsak sistemde bulunan bilgiyi gösterir.

**git status** --> git'in güncel durumunu gösterir.

**git init** --> bilgisayarınızda oluşturmuş olduğunuz bir klasörün Git'e tanımlanması için kullanılır.

**rm -rf .git** --> git'i silmemizi sağlar. Yalnız bu işlemi yaparken dosya yolunda git'in içinde olmalısın.

**git add dosya\_adi.uzanti** --> dosyayı index-staging kısmına alır yani commit yapmaya hazır hale getirir ama commit yapmaz.

**git add .** --> klasördeki her şeyi ekler.

**git commit** --> commit yapmamızı sağlar.

**git commit** -m "mesaj" --> commit atarken mesaj eklememizi sağlar.

**git log** --> commit'lerimizin tarihine, saatine ve kimin tarafından atıldığına bakarız.

**git remote add origin repo\_linki** --> git ile GitHub'ı bağlar.

**git push -u origin master** --> kodlarımızı push etmemizi sağlar.

**git push origin branch\_adi** --> yazdığın branch'e push eder.

**touch .gitignore** --> gitignore dosyasını oluştururuz sonra gitignore dosyasını açarız ve içerisine git'in görmesini istemediğimiz dosyaları veya klasörleri koyarız.

**git branch** --> içinde bulunduğumuz branch'i gösterir.

**git branch branch\_adi** --> branch oluşturmamızı sağlar.

**git switch branch\_adi** --> branch değiştirmemizi sağlar.

**git stash** --> ingilizcede “saklamak, depolamak” anlamında gelir. Üzerinde çalıştığınız fakat henüz commit etmediğimiz değişiklikleri geçici olarak kayıt altına alır.

**git stash pop** --> en son oluşturduğumuz fakat commit etmediğimiz kodlara geri dönmemizi sağlar

**git stash list** --> güncel stash'lerimizi gösterir.

**git stash clear** --> var olan bütün stash'leri siler.

**git restore dosya\_adi** --> dosyamızı bir önceki commit'deki haline çevirir.

**git chechkout** --> commit'lerimizin hash'lerini görüntülemizi sağlar.

**git reser hash\_kodu** --> commit silinir ama kodlarımız durur.

**git reset --hard hash\_kodu** --> commit ile birlikte kodlarımızı da siler

**git diff HEAD** --> güncel commit ile eski commit arasındaki farkı görürüz

**git diff hash\_kodu hash\_kodu** --> kıyaslamak istediğimiz 2 commit'e bakarız

kod çalışmazsa boşluk yerine yan yana 2 tane nokta koy

**git branch -r** --> remote branch'leri görüntüler.

**git checkout origin/master** --> remote branch'lere geçiş yapmamızı sağlar.

**git fetch origin master** --> fetch, Origin'den git master'ı bana getir. Değişikliklerimizi alır ve local'e getirir. Kodlarımız, dosyalarımızı vs. değiştirmediği için güvenli bir komuttur.

**git pull** --> git fetch + git merge, değişiklikleri al ama ayrı bir remote branch'te vs. tutma direk benim branch'ime göm. Dosyamızı ve local'imizi alıyo, bunları merge ve commit ediyo sonrada pull ile tüm değişiklikleri çekiyor.

**git clone repo\_linki** --> bulunduğun dosya yoluna, yazdığın linkin repo’sunu kopyalar.